A mixed-state CONDENSATION tracker with automatic model-switching

Michael Isard and Andrew Blake
University of Oxford, Oxford OX1 3PJ, UK.
misard@robots.ox.ac.uk, ab@robots.ox.ac.uk

Abstract

There is considerable interest in the computer vision community in representing and modelling motion. Motion models are used as predictors to increase the robustness and accuracy of visual trackers, and as classifiers for gesture recognition. This paper presents a significant development of random sampling methods to allow automatic switching between multiple motion models as a natural extension of the tracking process. The Bayesian mixed-state framework is described in its generality, and the example of a bouncing ball is used to demonstrate that a mixed-state model can significantly improve tracking performance in heavy clutter. The relevance of the approach to the problem of gesture recognition is then investigated using a tracker which is able to follow the natural drawing action of a hand holding a pen, and switches state according to the hand's motion.

1 Introduction

There is considerable interest in the computer vision community in representing and modelling motion [1, 3, 4]. Learned dynamical models, in the form of solutions to stochastic differential equations (SDEs), have been used [1] to improve the agility and robustness of visual trackers. There is a limit to the complexity of SDE based models, however, and a natural generalisation is to allow multiple models, with switching between models as appropriate. This allows a wider range of motion to be supported without losing the advantages of accurate prediction, and the model in use at a given time can also act as a recogniser discriminating between the distinct motions. A mixed discrete/continuous tracker combines the ability of continuous-valued motion models to track complex outlines with the powerful finite state-based systems used to describe sequential actions, for example in Hidden Markov Models. Kalman-filter based techniques to switch between multiple models are known in the control literature [2, 10]. In order to permit multiple models it is necessary to represent multiple hypotheses while motion is ambiguous. However, extensions of the Kalman filter to permit multiple hypotheses are known to face a combinatorial explosion.

Existing gesture recognition research, using either state-based [12, 3] or continuous-valued [5, 9] models divides the recognition process into two stages. First, some low-dimensional feature vector is extracted from an image (e.g., image-moments, or the output from a tracker). Only when this information has been extracted is recognition performed on the low-dimensional data. A great potential advantage of the multiple-model approach is that recognition and feature extraction are performed jointly, and so the form of the expected gesture can be used to guide feature search, potentially making it more efficient and robust. Existing gesture recognition systems also often work on rather coarse shapes, while some applications require a framework which allows detailed continuous-valued shape models rather than a finite number of possible object configurations.

Random sampling filters [7, 6, 8] were introduced to address the need to represent multiple hypotheses when tracking; the CONDENSATION algorithm [7] has been applied to the problem of visual tracking in clutter. This paper extends the CONDENSATION framework to permit a mixed-state object representation combining continuous-valued shape parameters with a discrete label encoding which one of a discrete set of motion models is in force. A joint p.d.f. for the mixed-state model is derived, and owing to the structure of the algorithm, model switching is performed jointly with tracking. Two benefits are therefore evident; tracking performance of certain classes of motions can be increased by building more appropriate models, and discrete parameters labelling the model-switching can be used as a source of information in their own right. The flexibility of the sampling framework allows a wide class of models to be used, which are not restricted to the solutions to SDEs.

2 The CONDENSATION Algorithm

A full description and derivation of the CONDENSATION algorithm is given in [7]. The filter's output at a given timestep t is an approximation of an entire probability distribution of likely object positions, represented as a weighted sample set $\{\mathbf{s}_t^{(n)},\ n = 1, \ldots, N\}$ with weights $\pi_t^{(n)}$. The iterative process applied to the sample-sets is shown in figure 1. At the top of the diagram, the output from timestep $t-1$ is the weighted sample-set $\{(\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}),\ n = 1, \ldots, N\}$. The first operation is to sample (with replacement) N times from the set $\{\mathbf{s}_{t-1}^{(n)}\}$, choosing a given element with probability $\pi_{t-1}^{(n)}$. Some elements may be chosen several times, while others may not be chosen at all.

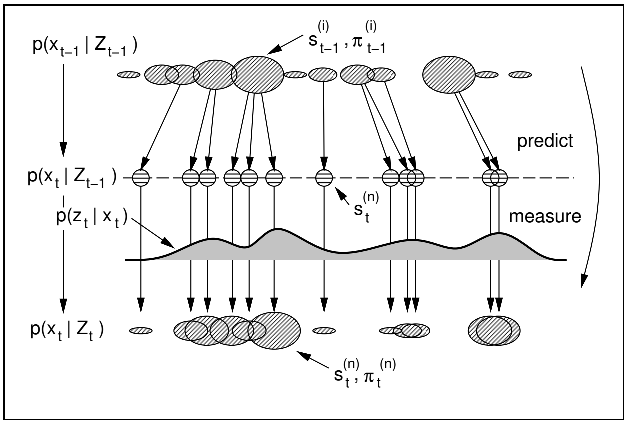


Figure 1: **One timestep in the CONDENSATION algorithm.** *Blob centres represent sample values and sizes depict sample weights.*

Each element chosen from the new set is now subjected to a predictive step which corresponds to sampling from the process density $p(\mathbf{X}_t | \mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{(i)})$ (where $\mathbf{X}_t$ is a parameter vector describing the object's state; its configuration and velocity at time t), so predictions from identical base samples will in general be different. Any dynamical model can be used within the algorithm provided the process density can be sampled. Finally the observation step is applied, calculating weights by evaluating the observation density $p(\mathbf{Z}_t|\mathbf{X}_t)$ to obtain the sample-set representation $\{(\mathbf{s}_t^{(n)}, \pi_t^{(n)})\}$ of the state-density for time t.

3 A mixed-state model

The fact that the process density $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ can have a somewhat general form can be exploited to allow the CONDENSATION algorithm to support, and automatically switch between, multiple motion models. The extended state is defined to be

$$\mathbf{X} = (\mathbf{x}, y),\ \mathbf{x} \in \mathbb{R}^{N_M},\ y \in \{1, \ldots, N_S\}$$

where y is a discrete variable labelling the current model, and $\mathbf{x}$ is a vector in the parameter space describing the object configuration and velocity. The process density can then be decomposed as follows:

$$p(\mathbf{X}_t|\mathbf{X}_{t-1}) = p(\mathbf{x}_t|y_t, \mathbf{X}_{t-1}) P(y_t|\mathbf{X}_{t-1})$$
$$P(y_t|\mathbf{X}_{t-1}):\ P(y_t = j|\mathbf{x}_{t-1}, y_{t-1} = i) = T_{ij}(\mathbf{x}_{t-1})$$

where the T_{ij} are state transition probabilities. The continuous motion models for each transition are given by the sub-process densities $p_{ij}(\mathbf{x}_t|\mathbf{x}_{t-1})$

$$p(\mathbf{x}_t|y_t, \mathbf{X}_{t-1}):\ p(\mathbf{x}_t|\mathbf{x}_{t-1}, y_{t-1} = i, y_t = j) = p_{ij}(\mathbf{x}_t|\mathbf{x}_{t-1}).$$

In order to implement a model in the CONDENSATION framework it is sufficient to specify a sampling algorithm for the process density, and the algorithm for mixed-state CONDENSATION is shown in figure 2. When tracking with a model of this form, discrete state transitions can be expected to occur automatically when appropriate. Each discrete state transition with non-zero probability contributes samples to the

Iterate

From the "old" sample set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, n = 1, \ldots, N\}$ at timestep $t-1$, construct a "new" sample set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)},\ n = 1, \ldots, N\}$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s}'^{(n)}_t = (\mathbf{x}'^{(n)}_t,\ i)$ as follows:
   (a) Generate a random number j with probability proportional to $\pi_{t-1}^{(j)}$. This is done efficiently by binary subdivision using cumulative probabilities.
   (b) set $\mathbf{s}'^{(n)}_t = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from $p(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{s}'^{(n)}_t)$ to choose $\mathbf{s}_t^{(n)} = (\mathbf{x}_t^{(n)},\ y_t^{(n)})$:
   (a) Sample transition probabilities
   $$P(y_t^{(n)} = j|\mathbf{X}_{t-1} = \mathbf{s}'^{(n)}_t) = T_{ij}(\mathbf{x}'^{(n)}_t)$$
   to find $y_t^{(n)}$.
   (b) Sample sub-process density
   $$p(\mathbf{x}_t^{(n)}|\mathbf{X}_{t-1} = \mathbf{s}'^{(n)}_t, y_t^{(n)} = j) = p_{ij}(\mathbf{x}_t|\mathbf{x}'^{(n)}_t)$$
   to find $\mathbf{x}_t^{(n)}$.
3. **Measure** and weight the new position in terms of the image data $\mathbf{Z}_t$:
   $$\pi_t^{(n)} = p(\mathbf{Z}_t|\mathbf{X}_t = \mathbf{s}_t^{(n)})$$

Finally normalise so that $\sum_n \pi_t^{(n)} = 1$.

Figure 2: **The mixed-state CONDENSATION algorithm.**

state distribution, and several such peaks are maintained while the motion is ambiguous. As soon as one model predicts significantly more accurately than the others, that model will dominate.

4 Tracking implementation

We follow standard methods to model both shape and continuous motion [1]. Object outlines are represented by quadratic B-splines, and the B-spline control points are restricted to lie in a linear subspace constructed using principal components analysis on a number of sample views. We assume that $T_{ij}(\mathbf{x}_t) \equiv T_{ij}$ and specify these probabilities by hand. Continuous motion is modelled as a second order SDE the parameters of which are learned from example training sequences using Maximum Likelihood Estimation [1]. Writing

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ x_{t-1} \end{pmatrix}$$

the required sampling scheme for an SDE-based subprocess density is given by

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + \begin{pmatrix} B\omega_t \\ 0 \end{pmatrix}, \quad \omega_t \in N(0,1)^{N_X}$$

where A and B are the learned parameters of the dynamical model and ω_t is a vector of i.i.d. random variables. When learning several sub-process densities for a variety of models, the training data is segmented by hand to distinguish the models. Future research will attempt to determine an automatic method for performing the segmentation of training data and learning T.

At each timestep, some output must be displayed to represent the estimated tracked position. While it would be feasible to compute the estimate as the weighted mean of the entire sample set, as in [7], it is expected that different discrete states will correspond to distinct clusters in configuration space, and so the following two stage approach is adopted. First the MAP estimate for the discrete state is found from

$$\begin{aligned} \hat{y}_t &= \arg\max_j P(y_t = j|\mathbf{Z}_1 \dots \mathbf{Z}_t) \\ &= \arg\max_j \sum_{n \in \Upsilon_j} \pi_t^{(n)}, \text{ where} \\ \Upsilon_j &= \{n \,|\, \mathbf{s}_t^{(n)} = (\mathbf{x}_t^{(n)}, j)\}. \end{aligned}$$

Then the estimate for the shape-space parameter vector is found from the weighted mean of that discrete sample set:

$$\begin{aligned} \hat{\mathbf{x}}_t &= \frac{\sum_{n \in \hat{\Upsilon}} \pi_t^{(n)} \mathbf{x}_t^{(n)}}{\sum_{n \in \hat{\Upsilon}} \pi_t^{(n)}} \\ \hat{\Upsilon} &= \{n \,|\, \mathbf{s}_t^{(n)} = (\mathbf{x}_t^{(n)}, \hat{y}_t)\} \end{aligned}$$

and it is this mean estimate which is displayed in later figures. The sample-set is initialised by determining the shape-space vector x_0 which corresponds to the initial position of the object in the sequence, and setting

$$\mathbf{x}_0^{(n)} = \begin{pmatrix} x_0 \\ x_0 \end{pmatrix}, \; y_0^{(n)} = 0, \; \pi_0^{(n)} = \frac{1}{N}, \text{ for all } n.$$

Automatic initialisation is of course of great interest, and is the subject of current research. The observation density $p(\mathbf{Z}|\mathbf{X})$ is chosen to suit the application as described in the following sections.

5 Tracking a bouncing ball

A model was constructed to track a ball which falls vertically and bounces on a table-top. This situation requires two discrete states where the second state corresponds to a "bounce event." This is a special form of continuous motion model which decays back to the default constant acceleration model after one timestep, giving the transition matrix (set manually)

$$T = \begin{pmatrix} 0.9 & 0.1 \\ 1.0 & 0.0 \end{pmatrix}$$

where the constant acceleration model is used for transitions $1 \to 1$ and $2 \to 1$ and the bounce model for transitions $1 \to 2$. The models, shown in figure 3, demonstrate the flexibility of the framework compared with an SDE-based approach. An affine shape-space $x = (x_1, h, x_3, \dots, x_6)^T$ is used where h is the ball's height. A random walk of small amplitude is used to model x_1 and the shape parameters $x_3 \dots x_6$. The parameters of the models are: constant acceleration due to gravity $a = 4.17\,\text{pixels/s}^2$, the standard deviation of vertical position noise for the constant acceleration model σ_h, the coefficient of restitution of the ball, $e = 0.67$, and the standard deviation of vertical position and velocity noise for the bounce model, respectively $\sigma_B = 2\,\text{pixels}$ and $\sigma_v = 10\,\text{pixels/s}$. The observation density $p(\mathbf{Z}_t|\mathbf{X}_t)$ used in this example follows established practice [7]. Normals of a fixed length γ are cast at M specified points s_m around the B-spline curve $\mathbf{r}(s)$ and edges are detected along those normals. Then

$$p(\mathbf{Z}|\mathbf{X}) \propto \exp\left\{ -\sum_{m=1}^{M} \frac{1}{2rM} f(\mathbf{z}_1(s_m) - \mathbf{r}(s_m); \gamma) \right\},$$

Constant acceleration model

$$h_t = h_{t-1} + v_{t-1} + a + \omega_t$$
$$v_t = h_t - h_{t-1}$$

$\omega_t \in N(0, \sigma_h)$

Bounce model

$$h_\tau = h_{t-1} + \tau h_{t-1}$$
$$v_\tau = v_{t-1} + \tau a$$
$$h_t = h_\tau - e(1-\tau)v_\tau + \gamma_t$$
$$v_t = -e v_\tau + (1-\tau)a + \nu_t$$

$\tau \in U[0,1),\ \nu_t \in N(0, \sigma_v),\ \gamma_t \in N(0, \sigma_B)$

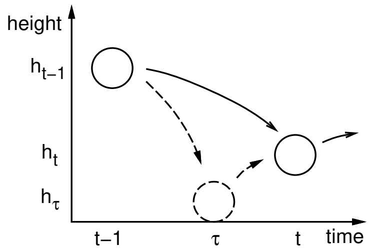
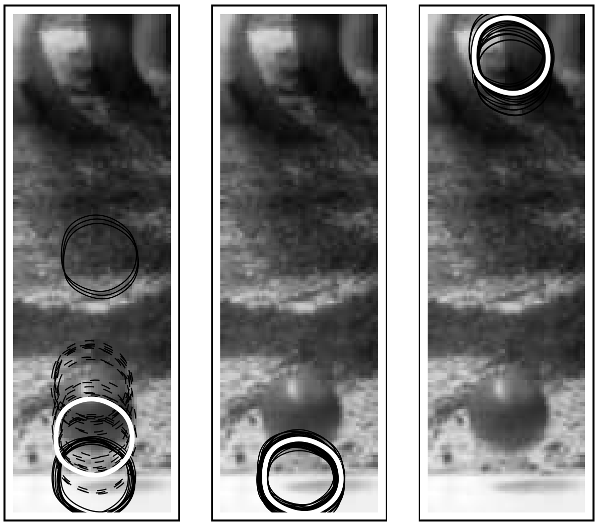


Figure 3: **Motion models for a bouncing ball.**

in which $r = 0.01$ is a variance constant, $\mathbf{z}_1(s)$ is the closest associated feature to $\mathbf{r}(s)$ and $f(\nu;\gamma) = \min(\nu^2, \gamma^2)$ to take account of the search scale γ.

When tracking against a blank background, the mixed-state model followed the bouncing ball as expected, and a single-state model with the same process noise failed at the first bounce. However, when the process noise for the single-state model was increased enough, it too tracked the bounces. The true utility of the more accurate mixed-state motion model is demonstrated when background clutter is added. Now the tracking problem becomes much harder, and a precisely tuned prediction is vital to prevent distractions by the clutter. A sequence was recorded, showing the ball bouncing in front of a highly cluttered backdrop (figure 4). The thick white outline shows the estimated ball position, while thin black outlines are high-scoring samples from a set of $N = 1500$. With $\sigma_h = 3$ pixels the mixed-state model tracked successfully (left, dashed outlines show samples which have performed a bounce transition). Although the estimate is misaligned slightly in this frame, enough samples are present in the vicinity of the ball that correct tracking continues. The single-state model with $\sigma_h = 3$ pixels tracked the ball on its initial descent, but at the moment of the first bounce tracking failed (middle). Increasing σ_h caused the single-state model to be distracted by clutter almost immediately (right).



$\sigma_h = 3$ pixels $\quad \sigma_h = 3$ pixels $\quad \sigma_h = 8$ pixels

Figure 4: **A mixed-state tracker outperforms a single-state tracker in clutter** *(see text).*

6 A three-state drawing model

Next, a tracker was constructed to follow the outline of a hand as it draws with a pen. Three distinct motions are included — a general line-drawing state, a stationary state, and a "scribbling" state corresponding to the rapid back-and-forth motion used when filling a region, which could perhaps be used in an interactive drawing package to signal that a flood-fill is required. An observation density was constructed for the hand tracker to take advantage of the known image properties of a hand drawing with black marker pen on a white page. The hand-coloured pixels are clustered around mid-grey, and the pixels on the page form two clusters, one around white and one around black. A distribution to represent this information consists of a single Gaussian $N(\mu_f, \sigma_f)$ for foreground pixels and a mixture $\alpha_1 N(\mu_{b1}, \sigma_{b1}) + \alpha_2 N(\mu_{b2}, \sigma_{b2})$ for the background. In all of the sequences used, the right edge of the hand was in slight shadow, so one set of coefficients was used for the left hand-edge and another for the right. The coefficients were set manually as shown in table 1. The observation density was then calculated as follows:

$$p(\mathbf{Z}|\mathbf{X}) \propto \prod_{m=1}^{M} y_m \left(\prod_{l=-\gamma}^{0} g_f(z_l(s_m)) \right) \left(\prod_{0}^{l=\gamma} g_b(z_l(s_m)) \right)$$

	μ_f	σ_f	α_1	μ_{b1}	σ_{b1}	α_2	μ_{b2}	σ_{b2}
unshadowed	120	14.1	1	60	20	1	225	20
shadowed	90	14.1	1	60	20	1	180	20

Table 1: **Gaussian coefficients for foreground and background pixels in hand images.** *Values are in units of grey-level intensity.*

where $z_l(s_m)$ is the greyscale intensity at distance l pixels along the normal to the curve at spline-parameter s_m (negative values indicate the interior of the object), g_f and g_b are the Gaussian mixtures outlined above, and y_m is a penalty constant which is set to 1 if an edge is detected in the direction of the normal at the position s_m and 0.3 otherwise. This penalty constant favours configurations which have detectable edges around the contour, but admits the possibility that clutter may locally have the same intensity as the hand, suppressing edge detection.

Six training sequences with approximately similar camera angles and lighting conditions, showing motions with increasing agility and against increasing clutter, were used to construct the models. Portions of each of the training sequences were hand-labelled as general motions or scribbling motions, and used to train SDE models. A PCA shape-space was built for the hand by first selecting 15 views and drawing around them interactively to create an initial shape-space, then using the tracker with that shape-space to find viewing angles of the hand which were not represented in the sample views. Next, 65 of these misaligned views were corrected manually and combined with the initial 15 views using PCA to form the final 12-dimensional shape-space. The transition probability matrix used was

$$T = \begin{pmatrix} 0.9800 & 0.0015 & 0.0185 \\ 0.0850 & 0.9000 & 0.0150 \\ 0.0050 & 0.0150 & 0.9800 \end{pmatrix}$$

which reflects the composition of a typical drawing. Most of the time is spent performing default motions $y = 1$, but there are occasional pauses $y = 2$ of short duration, and less frequently somewhat longer periods of scribbling $y = 3$. Also, scribbling motions often start or end with a pause.

Since the scribbling motion is an oscillator with small spatial extent, a variant of the standard SDE model was used which allows the mean of the oscillation to vary [11]. Each scribbling motion is considered to have a fixed mean, but distinct scribbles have distinct means. This is encoded by augmenting scribble-state samples with an extra vector denoting the mean configuration $\mathbf{X}_{\text{scribble}} = (\mathbf{x}, 3, \bar{x})$. The translation components of the mean vector $\bar{x}$ are initialised to be equal to the current position when a scribble begins (transition $1 \to 3$ or $2 \to 3$), and $\bar{x}$ is inherited from the previous sample over the course of that scribble (transition $3 \to 3$).

To test the tracker, a new sequence was recorded, which was not used to provide any training data. This was a 1250 field (25 s) sequence showing a drawing of a house (figure 5). Because of the high image velocities (up to 25 pixels/field) combined with the high dimensionality of the shape-space, $N = 15000$ samples per timestep were needed for robust tracking, which runs at approximately 0.33 Hz on an SGI O2 R5000 180MHz workstation. Tracking was accurate throughout, and sample frames are shown in the figure. The classification of motion by model-switching (figure 6) was also mostly accurate. The onset and end points of scribble gestures are found fairly reliably, although there is a slight lag in some of the switches which is to be expected since the motion is not unambiguous until at least a quarter of an oscillatory period has elapsed. Since discrete states are reported according to MAP estimates at single timesteps, information about the temporal coherence of discrete states is thrown away. More accurate model segmentation may be achievable by performing windowed smoothing over the data.

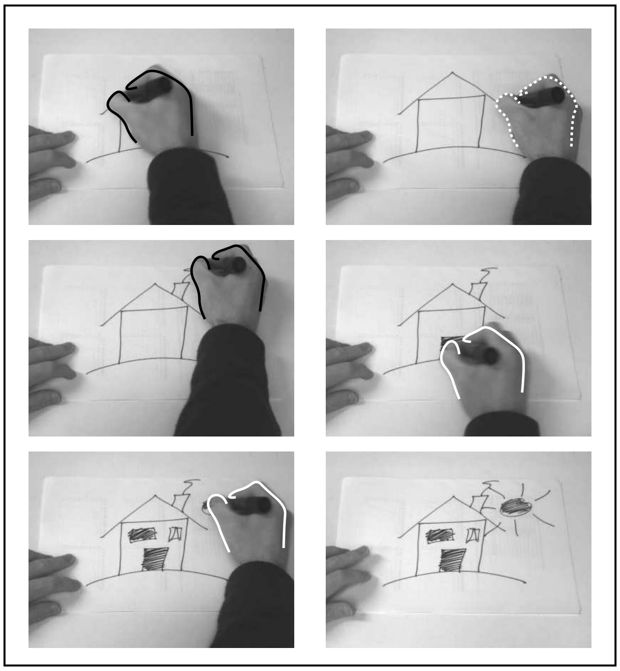


Figure 5: **A mixed-state tracker switches between models according to the motion of a drawing hand.** *The contour is drawn in black during default motions, white while scribbling, and dashed while stationary.*

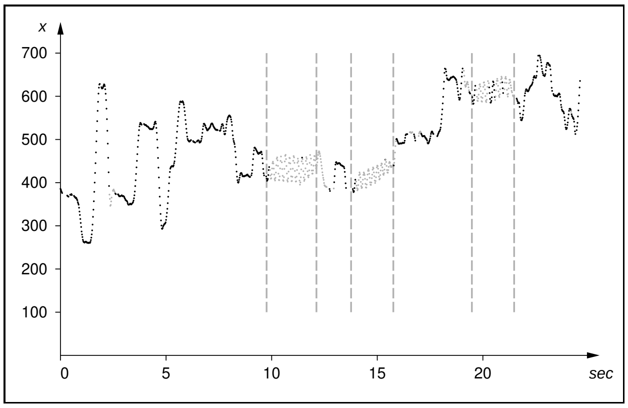


Figure 6: **Model switching provides a classification of motion.** *Black dots show drawing motion and grey dots scribbling. Stationary frames are not shown, for clarity. Dashed lines show the extents of the scribbling gestures as found by manual segmentation. The vertical axis shows x translation in pixels.*

Designing a visual tracker to drive a complex user-interface application is a very challenging problem, and there are still substantial obstacles to be overcome. First, a speedup by a factor of nearly 100 times is necessary; computing the observation density in hardware and switching to a multi-processor machine may bring this within reach. The system must also be able to determine when the pen is touching the paper, which may be achievable using stereo, or by analysing the hand's shadow on the page [13]. It is not clear how well the model-switching paradigm scales when more gestures are added, and how the tracker can be adapted for multiple users.

7 Conclusions

Just as interest in the study and characterisation of motion is growing, advances in filtering techniques offer new opportunities to experiment with much more flexible motion models than were previously available. In this paper a general class of motion representation — the mixed-state model — has been described, and a framework presented to implement such a model in a tracker. Mixed-state models have received relatively little attention in the computer-vision community due to the difficulty of incorporating them in traditional frameworks. The results presented demonstrate that mixed-state models can be used to improve tracking performance in the presence of complex motions, and that as a side-effect it is possible automatically to discriminate between distinct characteristic motions.

Several areas of research are suggested by this work. Learned motion models have previously been found [1] to be much more effective than hand-specified defaults, but learning a joint mixed-state model is an open research question. The sample-set size N needed to be rather large in experiments, running 10–100 times slower than real time; it may be possible to reduce it by improving the observation density, possibly by considering regions as well as contours. The problems of building a user-independent tracker and automatically initialising the algorithm have also still to be addressed. We acknowledge the support of the EPSRC.

References

[1] A. Blake, M.A. Isard, and D. Reynard. Learning to track the visual motion of contours. *Artificial Intelligence*, 78:101–134, 1995.

[2] Henk A. P. Blom and Yaakov Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783, August 1988.

[3] A.F. Bobick and A.D. Wilson. A state-based technique for the summarisation and recognition of gesture. In *Proc. 5th Int. Conf. on Computer Vision*, 382–388, 1995.

[4] C. Bregler and S.M. Omohundro. Nonlinear manifold learning for visual speech recognition. In *Proc. 5th Int. Conf. on Computer Vision*, 494–499, Boston, Jun 1995.

[5] C. J. Cohen, L. Conway, and D. Koditschek. Dynamical system representation, generation, and recognition of basic oscillatory motion gestures. In *Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition*, 151–156, October 1996.

[6] N. Gordon, D. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F*, 140(2):107–113, 1993.

[7] M.A. Isard and A. Blake. Visual tracking by stochastic propagation of conditional density. In *Proc. 4th European Conf. Computer Vision*, 343–356, Cambridge, England, Apr 1996.

[8] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

[9] S. Nagaya, S. Susumu, and R. Oka. A theoretical consideration of pattern space trajectory for gesture spotting recognition. In *Proc. 2nd Int. Conf. on Automatic Face and Gesture Recognition*, 151–156, October 1996.

[10] A. Pentland and A. Liu. Towards augmented control systems. In *IEEE Intelligent Vehicles*, 350–355, Detroit, MI, Sep 1995.

[11] D. Reynard, A.P. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motions from image sequences. In *Proc. 4th European Conf. Computer Vision*, 357–368, Cambridge, England, Apr 1996.

[12] T. Starner and A. Pentland. Visual recognition of american sign language using hidden markov models. In *Proc. Int. Workshop on Automated Face and Gesture Recognition*, 1995.

[13] J. Sullivan and A. Blake. Exploiting shadows in a visual, hand-driven user interface. In *Proc. British Machine Vision Conf.*, 1997.